NPS-MA-92-002

NAVAL POSTGRADUATE SCHOOL

Monterey, California

DOWNDATING OF SZEGO POLYNOMIALS AND DATA FITTING APPLICATIONS

W.B. Gragg
G.S. Ammar
L. Reichel

January 1992

Approved for public release; distribution unlimited
Prepared for: Naval Postgraduate School and the
National Science Foundation,
Washington D.C. 20550

NAVAL POSTGRADUATE SCHOOL
MONTEREY, CA 93943

Rear Admiral R. W. West, Jr.
Superintendent

Harrison Shull
Provost

This report was prepared in conjunction with research conducted for the National Science Foundation and for the Naval Postgraduate School Research Council. Funding was provided by the Naval Postgraduate School.

Prepared by:

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution unlimited
2b DECLASSIFICATION / DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S) NPS-MA-92-002	5 MONITORING ORGANIZATION REPORT NUMBER(S) NPS-MA-92-002
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b OFFICE SYMBOL (If applicable) MA	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School and the National Science Foundation
6c ADDRESS (City, State, and ZIP Code) Monterey, CA 93943	7b ADDRESS (City, State, and ZIP Code) Monterey, CA 93943 and Washington, D.C. 20550
8a NAME OF FUNDING / SPONSORING ORGANIZATION Naval Postgraduate School 8b OFFICE SYMBOL (If applicable) MA	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER O&MN, Direct funding
8c ADDRESS (City, State, and ZIP Code) Monterey, CA 93943	10 SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO / PROJECT NO / TASK NO / WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification)
DOWNDATING OF SZEGO POLYNOMIALS AND DATA FITTING APPLICATIONS

12 PERSONAL AUTHOR(S)
William Gragg, Greg Ammar, Lother Reichel

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Technical Report	FROM 1/91 TO 6/91	2 January 1992	

16 SUPPLEMENTARY NOTATION

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Szego polynomials, downdating, FFT, least squares, missing observations

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

Many algorithms for polynomial least squares approximation of real-valued function on a real interval determine polynomials that are orthogonal with respect to a suitable inner product defined on this interval. Analogously, it is convenient to compute Szegö polynomials, i.e., polynomials that are orthogonal with respect to an inner product on the unit circle, when approximating a complex-valued function on the unit circle in the least squares sense. It may also be appropriate to determine Szegö polynomials in algorithms for least squares approximation of real-valued periodic functions by trigonometric polynomials. This paper is concerned with Szegö polynomials that are defined by a discrete inner product on the unit circle. We present a scheme for downdating the Szegö polynomials and given least squares approximant when a node is deleted from the inner product. Our scheme uses the QR algorithm for unitary upper Hessenberg matrices. We describe a data-fitting application that illustrates how our scheme can be combined with the fast Fourier transform algorithm when the given nodes are not equidistant. Application to sliding windows is discussed also.

20 DISTRIBUTION / AVAILABILITY OF ABSTRACT	21 ABSTRACT SECURITY CLASSIFICATION	
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	UNCLASSIFIED	
22a NAME OF RESPONSIBLE INDIVIDUAL William Gragg	22b TELEPHONE (Include Area Code) (408) 646-2194	22c OFFICE SYMBOL MA/Gr

DD Form 1473, JUN 86 *Previous editions are obsolete*

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

ICM-9111-20

October 1991

Downdating of Szegö Polynomials and Data Fitting Applications

G.S. Ammar
Department of Mathematical Sciences
Northern Illinois University
DeKalb, IL 60115

W.B. Gragg
Department of Mathematics
Naval Postgraduate School Monterey, CA 93940

L. Reichel
Department of Mathematics and Computer Science
Kent State University
Kent, OH 44242-0001

ABSTRACT

Many algorithms for polynomial least squares approximation of real-valued function on a real interval determine polynomials that are orthogonal with respect to a suitable inner product defined on this interval. Analogously, it is convenient to compute Szegö polynomials, i.e., polynomials that are orthogonal with respect to an inner product on the unit circle, when approximating a complex-valued function on the unit circle in the least squares sense. It may also be appropriate to determine Szegö polynomials in algorithms for least squares approximation of real-valued periodic functions by trigonometric polynomials. This paper is concerned with Szegö polynomials that are defined by a discrete inner product on the unit circle. We present a scheme for downdating the Szegö polynomials and given least squares approximant when a node is deleted from the inner product. Our scheme uses the QR algorithm for unitary upper Hessenberg matrices. We describe a data-fitting application that illustrates how our scheme can be combined with the fast Fourier transform algorithm when the given nodes are not equidistant. Application to sliding windows is discussed also.

Downdating of Szegő Polynomials and Data Fitting Applications

G.S. Ammar* W.B. Gragg† L. Reichel‡

Abstract

Many algorithms for polynomial least squares approximation of a real-valued function on a real interval determine polynomials that are orthogonal with respect to a suitable inner product defined on this interval. Analogously, it is convenient to compute Szegö polynomials, i.e., polynomials that are orthogonal with respect to an inner product on the unit circle, when approximating a complex-valued function on the unit circle in the least squares sense. It may also be appropriate to determine Szegö polynomials in algorithms for least squares approximation of real-valued periodic functions by trigonometric polynomials. This paper is concerned with Szegö polynomials that are defined by a discrete inner product on the unit circle. We present a scheme for downdating the Szegö polynomials and given least squares approximant when a node is deleted from the inner product. Our scheme uses the QR algorithm for unitary upper Hessenberg matrices. We describe a data-fitting application that illustrates how our scheme can be combined with the fast Fourier transform algorithm when the given nodes are not equidistant. Application to sliding windows is discussed also.

1 Introduction

Let $\{z_k\}_{k=1}^m$ be a set of distinct nodes on the unit circle and let $\{w_k^2\}_{k=1}^m$ be a set of positive weights. Introduce for complex-valued functions g and h defined at the nodes the discrete inner product on the unit circle

$$(g,h)_m := \sum_{k=1}^{m} \overline{g(z_k)} h(z_k) w_k^2, \tag{1.1}$$

where the bar denotes complex conjugation. Polynomials that are orthogonal with respect to an inner product on the unit circle are known as *Szegő polynomials*. Let $\{\phi_j\}_{j=0}^{m-1}$ denote the family of *orthonormal* Szegő polynomials with respect to the inner product (1.1), where ϕ_j is of degree j and has a positive leading coefficient. The polynomials ϕ_j satisfy the *Szegő recurrence relations*

$$\sigma_0(z) = \bar{\phi}_0(z) = 1/\sigma_0,$$

*Department of Mathematical Sciences, Northern Illinois University, DeKalb, IL 60115.

†Department of Mathematics, Naval Postgraduate School, Monterey, CA 93940.

‡Department of Mathematics and Computer Science, Kent State University, Kent, OH 44242. Research supported by a National Research Council fellowship and NSF grant DMS-9002884.

$$\begin{aligned}\sigma_{j+1}\phi_{j+1}(z) &= z\phi_j(z) + \gamma_{j+1}\tilde{\phi}_j(z), \quad 0 \le j < m-1, \\ \sigma_{j+1}\tilde{\phi}_{j+1}(z) &= z\bar{\gamma}_{j+1}\phi_j(z) + \tilde{\phi}_j(z),\end{aligned} \tag{1.2}$$

where the recurrence coefficients $\gamma_{j+1} \in \mathbb{C}$ and $\sigma_{j+1} > 0$ are determined by

$$\begin{aligned}\sigma_0 &= \delta_0 = \left(\sum_{k=1}^{m} w_k^2\right)^{1/2}. \\ \gamma_{j+1} &= -(1, z\phi_j)_m / \delta_j, \\ \sigma_{j+1} &= \left(1 - |\gamma_{j+1}|^2\right)^{1/2}. \qquad 0 \le j < m. \\ \delta_{j+1} &= \delta_j \sigma_{j+1}.\end{aligned} \tag{1.3}$$

See, for example, Grenander and Szegö [4, Chapter 2]. It can be shown by induction that the auxiliary polynomials $\tilde{\phi}_j$ in (1.2) satisfy

$$\tilde{\phi}_j(z) = z^j \bar{\phi}_j(1/z), \quad 0 \le j < m,$$

where $\bar{\phi}_j(z)$ denotes the polynomial obtained by conjugating the coefficients of $\phi_j(z)$ in the power basis. Since the measure on the unit circle that defines (1.1) has precisely m points of increase, we have $|\gamma_j| < 1$ for $1 \le j < m$ and $|\gamma_m| = 1$. The coefficients γ_j are known as *Schur parameters*, and we refer to the σ_j as the associated *complementary parameters*. Although the complementary parameters are mathematically redundant, we retain them during computations in order to avoid numerical instability. Note from (1.3) that σ_0^2 is the total weight of the measure that defines (1.1), and that δ_j^{-1} is the leading coefficient of ϕ_j for $0 \le j < m$.

For later reference we also define the polynomial

$$\psi_m(z) := z\phi_{m-1}(z) + \gamma_m \tilde{\phi}_{m-1}(z). \tag{1.4}$$

Then

$$\psi_m(z) = \delta_{m-1}^{-1} \prod_{k=1}^{m} (z - z_k), \tag{1.5}$$

In particular, $(\phi_m, \phi_j) = 0$ for $0 \le j \le m$.

Example 1.1. Let $z_k := \exp(2\pi i(k-1)/m)$ and $w_k^2 := 1$ for $1 \le k \le m$, where $i := \sqrt{-1}$. Then $\sigma_0 = m^{1/2}$, $\sigma_j = 1$ and $\gamma_j = 0$ for $1 \le j < m$, and $\gamma_m = 1$. Thus, $\phi_j(z) = m^{-1/2} z^j$ for $0 \le j < m$, and $\psi_m(z) = m^{-1/2}(z^m - 1)$. □

Introduce the discrete norm

$$\|g\|_m := (g, g)_m^{1/2},$$

and let Π_{n-1} denote the set of all polynomials of degree at most $n-1$. Let f be a given complex-valued function defined at the nodes z_k, and consider the problem of computing the polynomial $p_{n-1} \in \Pi_{n-1}$, for some $n \leq m$, that satisfies

$$\|f - p_{n-1}\|_m = \min_{p \in \Pi_{n-1}} \|f - p\|_m . \tag{1.6}$$

The solution p_{n-1} of the minimization problem (1.6) can be expressed conveniently in terms of the Szegö polynomials ϕ_j as follows. Introduce the vector

$$\mathbf{f} = [w_1 f(z_1), w_2 f(z_2), \ldots, w_m f(z_m)]^T , \tag{1.7}$$

and the $m \times n$ matrix $Q = [q_{kj}]$,

$$q_{kj} := \phi_{j-1}(z_k) w_k , \quad 1 \leq k \leq m, \quad 1 \leq j \leq n, \tag{1.8}$$

where $w_k := \sqrt{w_k^2}$. Note that the matrix Q has orthonormal columns, i.e., $Q^* Q = I$, where $Q^* := \bar{Q}^T$. Let the vector $\mathbf{a} = [\alpha_1, \alpha_2, \ldots, \alpha_n]^T$ be defined by

$$\mathbf{a} := Q^* \mathbf{f} . \tag{1.9}$$

Then the polynomial p_{n-1} can be written as

$$p_{n-1} = \sum_{j=1}^{n} \alpha_j \phi_{j-1} , \tag{1.10}$$

where the Fourier coefficients α_j are independent of n.

It is the purpose of this paper to present an algorithm for *downdating* the recurrence coefficients γ_j and σ_j, as well as the Fourier coefficients α_j, when an node-weight pair is removed from the inner product (1.1). Our algorithm is based on the observation that the columns of the matrix Q are eigenvectors of a unitary Hessenberg matrix defined by the recurrence relations (1.2)–(1.3). This makes it possible to downdate the coefficients γ_j, σ_j and α_j by applying the QR algorithm for unitary Hessenberg matrices, presented in [5], with the node to be removed as shift. Details are described in Section 2.

We remark that the problem of *updating* the coefficients γ_j, σ_j and α_j when a node-weight pair $\{z_{m+1}, w_{m+1}^2\}$ is added to the inner product (1.1) is discussed in [8]. The updating problem can be solved by using an inverse QR algorithm for unitary Hessenberg matrices; see [8, 1].

Assume that we wish to determine the polynomial p_{n-1}, given by (1.9)–(1.10) with $n \leq m$ when the set of m nodes in the inner product (1.1) is a subset of the set of N equidistant

nodes $\{\exp(2\pi ij/N)\}_{j=0}^{N-1}$, with $\kappa := N - m > 0$ small. The weights w_k^2 are all assumed to be unity. Then it may be attractive to compute p_{n-1} by first computing the polynomial interpolant $p_{N-1} \in \Pi_{N-1}$ on the set of N equidistant nodes by using the fast Fourier transform (FFT) algorithm, and then determining p_{m-1} from p_{N-1} by applying our downdating scheme. The Fourier coefficients of the polynomial approximant p_{n-1} are then equal to the first n Fourier coefficients of p_{m-1}. This approach can similarly be applied to trigonometric polynomials. Details are described in Section 3. Computed examples are presented in Section 4 and Section 5 contains a summary.

Updating and downdating of polynomials approximants p_{n-1} when all the nodes z_k are *real* has received considerable attention in the literature; see Scott and Scott [9] and references there. A collection of algorithms for updating and downdating based on orthogonal polynomials is presented by Elhay et al. [3]. Algorithms for updating are also considered in [6, 7].

2 Downdating of Szegő polynomials

The connection between the Szegő polynomials determined by (1.1) and a unitary Hessenberg matrix can be seen as follows. Using the basis of Szegő polynomials, we can write

$$\sum_{l=1}^{j+1} \eta_{lj}\phi_{l-1}(z) = z\phi_{j-1}(z), \quad 1 \le j \le m, \tag{2.1}$$

where $\eta_{j+1,j} = \sigma_j$ for $1 \le j < m$, and $\eta_{m+1,m} = 1$. Let $\eta_{kj} := 0$ for $3 \le j+2 \le k \le m$, and define the upper Hessenberg matrix $H = [\eta_{kj}]_{j,k=1}^{m}$. Also define the unitary matrix $U = [\mu_{kj}]_{j,k=1}^{m}$ by

$$\mu_{kj} := \phi_{j-1}(z_k)w_k. \tag{2.2}$$

Substitution of $z = z_k$, $1 \le k \le m$, into (2.1) yields the equation

$$H^T U^T = U^T \Lambda, \tag{2.3}$$

or, equivalently,

$$H = U^T \Lambda U, \tag{2.4}$$

where $\Lambda := \operatorname{diag}[z_1, z_2, \ldots, z_m]$. Thus, H is a unitary upper Hessenberg matrix with positive subdiagonal elements that is uniquely determined by the inner product (1.1). Also note that the first n columns of U make up the matrix Q defined by (1.8). Algorithms for the solution of the least-squares problem (1.6) can therefore be viewed in terms of the spectral decomposition (2.4).

It is fairly straightforward to show, by using the recurrence relations (1.2)–(1.3), that H can be written as a product of m elementary unitary transformations that are defined by the recursion coefficients γ_j and σ_j for the ϕ_j. We have

$$H = G_1(\gamma_1)G_2(\gamma_2)\cdots G_{m-1}(\gamma_{m-1})G'_m(\gamma_m), \tag{2.5}$$

where the $G_j(\gamma_j)$, $1 \le j < m$, are $m \times m$ Givens matrices

$$G_j(\gamma_j) := \begin{bmatrix} I_{j-1} & & & \\ & -\gamma_j & \sigma_j & \\ & \sigma_j & \gamma_j & \\ & & & I_{m-j-1} \end{bmatrix}$$

and $G'_m(\gamma_m)$ is the diagonal matrix

$$G'_m(\gamma_m) := \operatorname{diag}[1, 1, \ldots, 1, -\gamma_m].$$

We refer to the representation (2.5) as the *Schur parametric form* of H.

The development of efficient algorithms for eigenproblems for unitary Hessenberg matrices is facilitated by the fact that every unitary upper Hessenberg matrix with nonnegative subdiagonal elements has a unique Schur parameterization. For example, when the implicitly shifted QR algorithm is applied to find the spectral decomposition of a unitary Hessenberg matrix, a sequence of intermediate unitary Hessenberg matrices is generated that converges to a diagonal matrix. In [5] the QR algorithm for unitary Hessenberg matrices is formulated in terms of the Schur parameters of the intermediate matrices. This results in an implementation that requires only $O(m)$ arithmetic operations per iteration on a matrix of order m.

Assume for the moment that the matrix H and scaling factor σ_0 are given, but that the nodes z_k and weights w_k^2 of the inner product (1.1) are not explicitly known. Then it follows from (2.4) and (2.2) that the nodes z_k are the eigenvalues of H, while the normalized weight w_k/σ_0 is equal to the first component of the normalized eigenvector corresponding with z_k. (We assume that each eigenvector is scaled to have unit Euclidean length and a nonnegative first component.) The unitary Hessenberg QR algorithm can be used to compute the matrix Λ and vector $U\mathbf{e}_1$, without computing the rest of U, with not more than $O(m)$ arithmetic operations per iteration. Throughout this paper $\mathbf{e}_j$ denotes the j^{th} axis vector in $\mathbb{C}^m$. The QR algorithm determines one node-weight pair at a time, and for each pair computed the order of the unitary

upper Hessenberg matrix is reduced by one, so that the reduced Hessenberg matrix corresponds with the node-weight pairs that have not yet been determined.

We remark that in the case that the nodes of the discrete inner product are real, then the analogue of the matrix H is a real symmetric tridiagonal matrix T with positive subdiagonal elements. This matrix T contains recurrence coefficients for orthonormal polynomials that satisfy a three-term recurrence relation. Golub and Welsch [4] proposed the use of the QR algorithm for symmetric tridiagonal matrices for the computation of the node-weight pairs associated with T. This algorithm also determines one node-weight pair at a time, and reduces the order of T when such a pair has been found.

Conversely, the construction of H from the inner product (1.1) can be regarded as an inverse eigenvalue problem. In particular, given the matrix $\Lambda = \mathrm{diag}[z_1, z_2, \ldots, z_m]$ and the vector $\mathbf{q}_0 := \sigma_0^{-1}[w_1, w_2, \ldots, w_m]^T$, we can perform a sequence of elementary unitary similarity transformations whose composition results in an $m \times m$ unitary matrix U such that the matrix on the right-hand of

$$\begin{bmatrix} 1 & 0^T \\ 0 & U^* \end{bmatrix} \begin{bmatrix} \star & \mathbf{q}_0^* \\ \mathbf{q}_0 & \Lambda \end{bmatrix} \begin{bmatrix} 1 & 0^T \\ 0 & U \end{bmatrix} = \begin{bmatrix} \star & \sigma_0 \mathbf{e}_1^T \\ \sigma_0 \mathbf{e}_1 & H \end{bmatrix} \tag{2.6}$$

is of upper Hessenberg form with positive subdiagonal elements. (The $\star$ represents an arbitrary scalar that remains unchanged.) In other words, $U^* \mathbf{q}_0 = \sigma_0 \mathbf{e}_1$, and $U^* \Lambda U$ is a unitary upper Hessenberg matrix H. Consequently, H is the matrix corresponding with the inner product (1.1).

The transformation of Λ to H can be performed using $O(m^2)$ arithmetic operations with the inverse unitary QR (IUQR) algorithm described in [1]. The IUQR algorithm is an updating procedure because it incorporates node-weight pairs one at a time. After the j^{th} stage of the algorithm has been executed, the $j \times j$ unitary Hessenberg matrix corresponding with the inner product determined by the first j nodes and weights has been obtained.

In [8] the approximation problem (1.6) is solved using the IUQR algorithm to construct the Schur parameters of the unitary Hessenberg matrix H. The elementary unitary matrices are accumulated against the vector $\mathbf{f}$ during the algorithm to obtain the vector of Fourier coefficients $\mathbf{a} = U^* \mathbf{f}$ without explicitly forming U. In this way we obtain the interpolating polynomial p_{m-1} that is the solution of (1.6) with $n = m$ in $O(m^2)$ arithmetic operations. Of course, the partial sums of the Fourier expansion of the interpolating polynomial yields the solution (1.10) of (1.6) for each $n < m$. Moreover, if one is interested in computing only p_{n-1}

for $n < m$, then the algorithm can be *curtailed* so that only $O(mn)$ arithmetic operations are required for the computation of the parameters $\{\gamma_j\}_{j=1}^{n}$, $\{\sigma_j\}_{j=0}^{n}$, and $\{\alpha_j\}_{j=1}^{n}$. See [8] for details.

Now assume that we have solved the least-squares problem (1.6) with $n = m$ by the method described in [8], so that sets of Schur parameters $\{\gamma_j\}_{j=1}^{m}$ and of complementary parameters $\{\sigma_j\}_{j=0}^{m}$ corresponding with the inner product (1.1), as well as sets of Fourier coefficients $\{\alpha_j\}_{j=1}^{m}$ of the interpolating polynomial p_{m-1} are explicitly known. In the downdating problem, we seek to solve the corresponding least-squares problem with one term deleted from (1.1). In particular, let $\hat{z}$ be the node that is to be deleted from the inner product (1.1) and let $\hat{w}$ be the square-root of the associated weight. In order to simplify some formulas that follow, we assume without loss of generality that $\hat{z} = z_m$ and $\hat{w} = w_m$.

Introduce the inner product

$$\langle g, h \rangle_{m-1} := \sum_{k=1}^{m-1} \overline{g(z_k)}\, h(z_k)\, w_k^2 \tag{2.7}$$

and discrete norm

$$\|g\|_{m-1} := \langle g, g \rangle_{m-1}^{1/2}.$$

We seek the polynomial $\hat{p}_{m-2} \in \Pi_{m-2}$ such that

$$\|f - \hat{p}_{m-2}\|_{m-1} = \min_{p \in \Pi_{m-2}} \|f - p\|_{m-1}. \tag{2.8}$$

Denote the family of orthonormal Szegő polynomials with positive leading coefficient associated with (2.7) by $\{\hat{\phi}_j\}_{j=0}^{m-2}$. Also let $\{\hat{\gamma}_j\}_{j=1}^{m-1}$ and $\{\hat{\sigma}_j\}_{j=0}^{m-1}$ denote the sets of recurrence coefficients for the $\hat{\phi}_j$, and let $\hat{\Lambda} := \mathrm{diag}[z_1, \ldots, z_{m-1}]$ and $\hat{\mathbf{q}}_0 := \hat{\sigma}_0^{-1}[w_1, \ldots, w_{m-1}]^T$, where $\hat{\sigma}_0 := (\sigma_0^2 - \hat{w}^2)^{1/2}$ is the total weight of the measure that defines (2.7). Then from the above discussion, there is a unique unitary Hessenberg matrix $\hat{H}$ and a unique unitary matrix $\hat{U}$ such that

$$\hat{U}\,\hat{\mathbf{q}}_0 = \hat{\sigma}_0 \hat{\mathbf{e}}_1$$

and

$$\hat{H} = \hat{U}^* \hat{\Lambda} \hat{U}, \tag{2.9}$$

where $\hat{\mathbf{e}}_j$ denotes the j^{th} axis vector in $\mathbb{C}^{m-1}$. Moreover, the recurrence coefficients $\hat{\gamma}_j$ and $\hat{\sigma}_j$ are the Schur parameters and complementary parameters, respectively, of $\hat{H}$. The optimal polynomial is then given by

$$\hat{p}_{m-2}(z) = \sum_{k=1}^{m-1} \alpha_k \phi_{k-1}(z).$$

where the vector of Fourier coefficients $\hat{\mathbf{a}} = [\hat{a}_1, \hat{a}_2, \ldots, \hat{a}_{m-1}]^T$ is given by $\hat{\mathbf{a}} := \hat{U}^* \hat{\mathbf{f}}$ and $\hat{\mathbf{f}} := [w_1 f(z_1), \ldots, w_{m-1} f(z_{m-1})]^T$.

Our scheme for downdating is based on the observation that $\hat{H}$ and $\hat{U}$ can be computed from H and U by applying one step of the QR algorithm with "ultimate" shift $\hat{z}$, together with some permutation similarity transformations, to determine a unitary matrix W such that

$$\begin{bmatrix} 1 & 0^T \\ 0 & W^* \end{bmatrix} \begin{bmatrix} \star & \sigma_0 \mathbf{e}_1^T \\ \sigma_0 \mathbf{e}_1 & H \end{bmatrix} \begin{bmatrix} 1 & 0^T \\ 0 & W \end{bmatrix} = \begin{bmatrix} \star & \hat{\sigma}_0 \hat{\mathbf{e}}_1^T & \hat{w} \\ \hat{\sigma}_0 \hat{\mathbf{e}}_1 & \hat{H} & 0 \\ \hat{w} & 0^T & \hat{z} \end{bmatrix}. \tag{2.10}$$

Then

$$\begin{bmatrix} \hat{U} & 0 \\ 0^T & 1 \end{bmatrix} = UW$$

and

$$\begin{bmatrix} \hat{\mathbf{a}} \\ \hat{w} f(\hat{z}) \end{bmatrix} = W^* \mathbf{a} . \tag{2.11}$$

Thus, the downdated Fourier coefficients $\hat{a}_k$ are obtained by accumulating each elementary unitary transformation against $\mathbf{a}$. An efficient implementation is obtained by using the QR algorithm for unitary Hessenberg matrices described in [5].

We now assume that $\hat{z} := z_l$ for some l, $1 \le l \le m$, and let $\hat{w} := w_l$. One step of the QR algorithm with shift $\hat{z}$ applied to the matrix H determines a unitary upper Hessenberg matrix $\tilde{V}$, such that

$$H' := \tilde{V}^* H \tilde{V} = \begin{bmatrix} \tilde{H} & 0 \\ & \\ 0^T & \hat{z} \end{bmatrix},$$

because $\hat{z}$ is an eigenvalue of H. It is easy to see, however, that the QR algorithm applied to H will not yield the required $\hat{H}$, because the vector $\sigma_0 \mathbf{e}_1$ will not be transformed as required by (2.10). On the other hand, an RQ algorithm for H, in which the transforming matrix $\tilde{V}$ would be a product of Givens matrices in the reverse order of (2.12) below, would yield the desired similarity transformation.

Instead of modifying the unitary Hessenberg QR algorithm of [5] to perform an RQ iteration, we apply the QR algorithm to the unitary Hessenberg matrix $H^P := JH^TJ$, where $J = [\mathbf{e}_m, \mathbf{e}_{m-1}, \ldots, \mathbf{e}_1]$ is the *reversal matrix* of order m. It is easily seen that if H is given by (2.5), then

$$H^P = G_1(\bar{\gamma}_1) G_2(\bar{\gamma}_2) \cdots G_{m-1}(\bar{\gamma}_{m-1}) G'_m(\bar{\gamma}_m),$$

where $\bar{\gamma}_j := \bar{\gamma}_{m-j} \gamma_m$ for $1 \le j < m$ and $\bar{\gamma}_m := \gamma_m$. The application of the QR algorithm on H^P is equivalent with that of the RQ algorithm on H. One iteration of the QR algorithm with

shift $\hat{z}$ applied to H^P generates a unitary upper Hessenberg matrix

$$V = G_1(\delta_1)G_2(\delta_2)\cdots G_{m-1}(\delta_{m-1})G'_m(\delta_m) \tag{2.12}$$

such that

$$V^* H^P V = \begin{bmatrix} H' & 0 \\ 0^T & \hat{z} \end{bmatrix} . \tag{2.13}$$

Moreover, δ_m can be taken to be an arbitrary unimodular number because deflation has taken place [5]. Let $\tau_j := (1 - |\delta_j|^2)^{1/2}$ denote the complementary parameter to δ_j for $1 \le j \le m$.

Observe that only the last two components of $V^*\sigma_0 \mathbf{e}_m$ are nonzero, and that $|\delta_m| = 1$ can be chosen so that these two components are given by

$$\begin{bmatrix} 1 & 0 \\ 0 & -\delta_m \end{bmatrix} \begin{bmatrix} \delta_{m-1} & \tau_{m-1} \\ \tau_{m-1} & \bar{\delta}_{m-1} \end{bmatrix} \begin{bmatrix} 0 \\ \sigma_0 \end{bmatrix} = \begin{bmatrix} \tau_{m-1}\sigma_0 \\ |\delta_{m-1}|\sigma_0 \end{bmatrix} .$$

Finally, we transform the right-hand side of (2.13) by similarity using $\begin{bmatrix} \hat{J} & 0 \\ 0^T & 1 \end{bmatrix}$, where $\hat{J}$ is the reversal matrix of order $m-1$, and transpose the result. In this way we transform H to $W^* H W = \begin{bmatrix} H'' & 0 \\ 0^T & \hat{z} \end{bmatrix}$, where $W = JV \begin{bmatrix} \hat{J} & 0 \\ 0^T & 1 \end{bmatrix}$. Moreover, $W^*\sigma_0 e_1 = \begin{bmatrix} \sigma_0\tau_{m-1}\hat{\mathbf{e}}_1 \\ |\delta_{m-1}|\sigma_0 \end{bmatrix}$, and by the uniqueness of the reduction, $H'' = \hat{H}$, $\sigma_0\tau_{m-1} = \hat{\sigma}_0$, and $|\delta_{m-1}|\sigma_0 = \hat{w}$. The vector of Fourier coefficients $\hat{\mathbf{a}}$ is then determined from (2.11) by applying W^* to $\mathbf{a}$ incrementally.

Observe that our downdating procedure requires knowledge of the node to be deleted but not of the corresponding weight. We can therefore compare the computed value of $\hat{w}$ to the actual value w_l in order to assess the accuracy of the computations. If $\hat{w}$ is not close to w_l, then the downdated polynomials $\hat{\phi}_j$ and $\hat{p}_{m-2}$ might not be accurate. Another accuracy check is provided by the computed value of ρ_m in the algorithm. This quantity is the m^{th} diagonal element of the upper triangular matrix in the QR factorization of $H^P - \hat{z}I$, which is mathematically zero when $\hat{z}$ is an eigenvalue of H. If the computed value of ρ_m is not "tiny", then the computed downdated polynomial might be inaccurate. We therefore consider ρ_m and $\hat{w}$ as being part of the output of the algorithm.

Algorithm 2.1 (Downdating by removing the node-weight pair $\{\hat{z}, \hat{w}\}$)

Input: m, $\{\gamma_j\}_{j=1}^{m}$, $\{\sigma_j\}_{j=0}^{m}$, $\{\alpha_j\}_{j=1}^{m}$, $\hat{z}$;

Output: $\{\hat{\gamma}_j\}_{j=1}^{m-1}$, $\{\hat{\sigma}_j\}_{j=0}^{m-1}$, $\{\hat{\alpha}_j\}_{j=1}^{m-1}$, ρ_m, $\hat{w}$;

$[\gamma_j]_{j=1}^{m-1} := \gamma_m[\bar{\gamma}_{m-j}]_{j=1}^{m-1}$; $[\sigma_j]_{j=1}^{m-1} := [\sigma_{m-j}]_{j=1}^{m-1}$;

$[\alpha_j]_{j=1}^{m} := [\alpha_{m-j+1}]_{j=1}^{m}$;

$\delta_0 := \bar{\delta}_0 := 1$;

for $j := 1.2, \ldots, m-1$ do

$\quad \rho_j := \left(\sigma_j^2 + |\hat{z}\delta_{j-1} + \gamma_j\bar{\delta}_{j-1}|^2\right)^{1/2}$;

$\quad$ if $j \geq 2$ then $\hat{\sigma}_{j-1} := \tau_{j-1}\rho_j$;

$\quad \tau_j := \sigma_j/\rho_j$;

$\quad \delta_j := (\hat{z}\delta_{j-1} + \gamma_j\bar{\delta}_{j-1})/\rho_j$;

$\quad \hat{\alpha}_j := -\delta_j\alpha_j + \tau_j\alpha_{j+1}$;

$\quad \alpha_{j+1} := \tau_j\alpha_j + \bar{\delta}_j\alpha_{j+1}$;

$\quad \bar{\delta}_j := (\bar{\delta}_{j-1} + \bar{\gamma}_j\hat{z}\delta_{j-1})/\rho_j$;

$\quad \hat{\gamma}_j := \delta_j\bar{\delta}_j - \tau_j^2\gamma_{j+1}\bar{z}_l$;

$\rho_m := |\hat{z}\delta_{m-1} + \gamma_m\bar{\delta}_{m-1}|$;

$\hat{\sigma}_{m-1} := \tau_{m-1}\rho_m$; $\hat{w} = |\delta_{m-1}|\sigma_0$; $\hat{\sigma}_0 = \tau_{m-1}\sigma_0$;

$\hat{\gamma}_{m-1} := \hat{\gamma}_{m-1}/|\hat{\gamma}_{m-1}|$; $\hat{\sigma}_{m-1} := 0$; (parameter correction)

$[\hat{\gamma}_j]_{j=1}^{m-2} := \hat{\gamma}_{m-1}[\bar{\hat{\gamma}}_{m-j-1}]_{j=1}^{m-2}$; $[\hat{\sigma}_j]_{j=1}^{m-2} := [\hat{\sigma}_{m-j-1}]_{j=1}^{m-2}$;

$[\hat{\alpha}_j]_{j=1}^{m-1} := [\hat{\alpha}_{m-j}]_{j=1}^{m-1}$; □

The algorithm overwrites the Fourier coefficients $\{\alpha_j\}_{j=1}^{m}$ with intermediate quantities. Algorithm 2.1 requires $O(m)$ arithmetic operations $(+, -, *, /)$ and the evaluation of $m-1$ square roots.

We have already noted that if the nodes z_k are real, then the analogue of the matrix H is a real symmetric tridiagonal matrix T with positive subdiagonal elements. Similarly with Algorithm 2.1, downdating of the orthonormal polynomials associated with T, as well as of Fourier coefficients, can be carried out by algorithms based on the QR algorithm for symmetric tridiagonal matrices. This observation may be new.

In certain data-fitting applications it may be desirable to update the polynomial p_{m-1}, given by (1.9)–(1.10) with $n = m$, by replacing certain node-weight pairs. This can be carried out by successively removing an node-weight pair by Algorithm 2.1, and then adding a new

node-weight pair using one step of Algorithm 3.1 in [8]. This combination of algorithms yields a sliding window scheme.

3 Downdating and the FFT algorithm

When the nodes z_k are equidistant on the unit circle and all weights w_k^2 are unity, then interpolation and least-squares approximation of a given function by algebraic or trigonometric polynomials can be carried out rapidly by the FFT algorithm. This section describes in some detail how our downdating scheme may be combined with the FFT algorithm to achieve rapid schemes for interpolation when the nodes are essentially equidistant. More precisely, we will consider the case when the set of nodes $\{z_k\}_{k=1}^m$ in the inner product (1.1) is a subset of the set of equidistant nodes $\{\exp(2\pi ij/N)\}_{j=0}^{N-1}$, where $\kappa := N - m$ is a small positive integer. In our operation count we will assume that κ is independent of m. The weights w_k^2 in (1.1) are all assumed to be unity.

Let f denote a complex-valued function, whose values $f(z_k)$, $1 \le k \le m$, are explicitly known. We remark that a representation of the interpolating polynomial $p_{m-1} \in \Pi_{m-1}$ in Newton form can be computed in $O(m^2)$ arithmetic operations. The Vandermonde solver by Björck and Pereyra [2] can be used to determine a representation of p_{m-1} in terms of the monomial basis, and requires also $O(m^2)$ arithmetic operations. Our scheme only requires $O(m \log m)$ arithmetic operations and yields a representation of p_{m-1} in the basis of Szegö polynomials.

Let $\{z_k'\}_{k=1}^{\kappa}$ denote the complement of the set $\{z_k\}_{k=1}^m$ in $\{\exp(2\pi ij/N)\}_{j=0}^{N-1}$, and let $f(z_k') := 0$, $1 \le k \le \kappa$. Thus, f is defined at the N roots of unity $\exp(2\pi i(j-1)/N)$, $1 \le j \le N$, and we can compute the Fourier coefficients of the polynomial $p_{N-1} \in \Pi_{N-1}$ that interpolates f at these nodes by the FFT algorithm in $O(N \log N) = O(m \log m)$ arithmetic operations. Using the Schur parameters given in Example 1.1, we apply Algorithm 2.1 κ times to eliminate the nodes z_k', $1 \le k \le \kappa$, from the inner product. This requires $O(m)$ arithmetic operations and yields the Fourier coefficients of the desired interpolating polynomial p_{m-1}. The Fourier coefficients of the least squares approximants $p_{n-1} \in \Pi_{n-1}$ with $n < m$ are, of course, a subset of the Fourier coefficients of p_{m-1}.

A scheme closely related to the one outlined above can be used to compute trigonometric polynomials rapidly. Let z_k and z_k' be the nodes introduced above, and define $\theta_k := \arg(z_k)$, $1 \le k \le m$, and $\theta_k' := \arg(z_k')$, $1 \le k \le \kappa$. Also assume that m is odd, i.e., $m = 2r + 1$. Let

$f(\theta)$ be a real-valued function defined at the nodes θ_k and let $f(\theta'_k) := 0$, $1 \le k \le \kappa$. We wish to compute a trigonometric polynomial $t(\theta) \in \text{span}\{1, \cos\theta, \ldots, \cos r\theta, \sin\theta, \ldots, \sin r\theta\}$ that minimizes the sum

$$\sum_{k=1}^{m} (f(\theta_k) - t(\theta_k))^2 .$$

This can be accomplished as follows. First solve the minimization problem

$$\min_{\alpha_j \in \mathbb{C}} \sum_{k=1}^{N} |e^{2\pi i(k-1)r/N} f\left((k-1)\frac{2\pi}{N}\right) - \sum_{j=1}^{N} \alpha_j e^{2\pi i(k-1)(j-1)/N}|^2$$

by the FFT algorithm in $O(m \log m)$ arithmetic operations, and denote the optimal coefficients by $\tilde{\alpha}_j$. Remove the nodes θ'_k, $1 \le k \le \kappa$, by applying the downdating scheme κ times to the polynomial $\tilde{p}_{2r+\kappa}(z) := \sum_{j=0}^{2r+\kappa} \tilde{\alpha}_j z^j$, where $N - 1 = 2r + \kappa$. This requires $O(m)$ arithmetic operations and yields the polynomial $p_{2r} \in \Pi_{2r}$. The desired trigonometric polynomial is then given by $t(\theta) := z^{-r} p_{2r}(z)$, $z = \exp(i\theta)$.

4 Computed examples

We now present the results of some numerical experiments with our downdating procedure. These experiments were performed in FORTRAN on a SparcStation SLC at Northern Illinois University, on which there are approximately 7 and 16 significant decimal digits in single-precision and double-precision calculations, respectively.

The first set of experiments compares the accuracy of the downdating procedure with that of the updating procedure IUQR described in [8]. We input N unimodular nodes $\{z_j\}_{j=1}^{N}$, positive weights $\{w_j^2\}_{j=1}^{N}$, and complex function values $\{f(z_j)\}_{j=1}^{N}$. For any positive integer $m \le N$, let $\mathbf{a}_m$ denote the vector of Fourier coefficients of the solution p_{m-1} of (1.6) with $n = m$, and let $\mathbf{g}_m$ denote the vector of Schur parameters determined by the inner product (1.1) and recurrence relations (1.2)–(1.3).

We first compute vectors $\tilde{\mathbf{a}}_N$ and $\tilde{\mathbf{g}}_N$ using an implementation of the IUQR algorithm in single-precision arithmetic [8]. We then repeatedly apply a single-precision implementation of Algorithm 2.1 to compute $\tilde{\mathbf{a}}_m$ and $\tilde{\mathbf{g}}_m$ for decreasing values of m. The k^{th} application of Algorithm 2.1 removes the node z_{N-k+1} from the inner product to compute the solution of (1.6) with $n := m := N - k$. After each downdating step, we calculate the relative error in $\tilde{\mathbf{a}}_m$, i.e., $\|\mathbf{a}_m - \tilde{\mathbf{a}}_m\|_2 / \|\mathbf{a}_m\|_2$, and the error in $\tilde{\mathbf{g}}_m$, i.e., $\|\mathbf{g}_m - \tilde{\mathbf{g}}_m\|_2$, where $\|\mathbf{x}\|_2$ denotes the Euclidean norm of $\mathbf{x} \in \mathbb{C}^m$. We also solve each problem of order m using the IUQR algorithm in single-precision arithmetic and compute the resulting errors. The results of the IUQR algorithm in

double-precision arithmetic are used as exact answers in the error calculations. The following tables display the resulting errors for the downdating procedure (DD) and the updating procedure (UD). In all of the experiments, each function value $f(z_j)$ has its real part and imaginary part randomly generated according to a uniform distribution in $[-5, 5]$.

We first choose the nodes to be the $N := 300$ roots of unity $z_j := \exp(2\pi i(j-1)/N)$, $1 \le j \le N$, and uniform weights $w_j^2 := 1$. Table 1 shows the errors for problems of order $m := N - k$ for various values of k. Table 1 also shows the results with the same choice of nodes and weights, except that the nodes are permuted in a random way. This permutation changes the nodes that are deleted as well as the order in which the nodes are added in the updating procedure. It should be noted that the errors in the downdating procedure can be expected to increase as k increases. Table 2 shows that similar results are obtained with the same set of nodes and randomly generated weights in $(0, 10)$.

Table 3 shows the results obtained with uniform weights and $N := 300$ nodes $z_j := \exp(\pi i(j-1)/N)$, $1 \le j \le N$, in $[0, \pi)$, both in their original order and in a random order. Here again, the downdating procedure performs well.

Table 4 shows the results when the initial 300 nodes are randomly selected points on the unit circle. In this example, the error in the downdating procedure displays a sudden increase at $k = 10$. In this step the error in the computed downdated weight, $|w - \tilde{w}|$, was greater than 10^{-1}. Observe that the error incurred at this downdating step propagated to the subsequent downdating steps, but the errors in $\tilde{\mathbf{a}}_m$ and $\tilde{\mathbf{g}}_m$ seem to grow gradually as k increases, i.e., as $m := N - k$ decreases. Other experiments with random nodes produced similar results. It should be noted that in our experiments, a large error in the computed weight did not always coincide with a large jump in the errors in $\tilde{\mathbf{a}}_m$ and $\tilde{\mathbf{g}}_m$. It is clear that more work needs to be done in order to understand the numerical aspects of the updating and downdating problems.

Our final experiment tests the accuracy and speed of the procedure described in Section 3 for downdating the FFT. The N-point FFT is used to obtain the Fourier coefficients of the interpolating polynomial p_{N-1}. A randomly selected set of nodes is then removed from the inner product using Algorithm 2.1. This experiment was run with a radix-2 FFT subroutine in single precision arithmetic with $N := 1024$ and $N := 2048$. Table 5 shows the computed error after k downdating steps for various values of k. As above, we use the results of the IUQR algorithm in double precision arithmetic as exact answers for error checking. We also display the amount of CPU time required by the FFT with k downdating steps and the time required

Table 1: 300 nodes with equispaced arguments in $[0, 2\pi)$; uniform weights.

	nodes in order of increasing argument.				nodes in random order.			
	relative error in $\tilde{a}_m$		error in $\tilde{g}_m$		relative error in $\tilde{a}_m$		error in $\tilde{g}_m$	
k	DD	UD	DD	UD	DD	UD	DD	UD
0	0.122E-03	0.122E-03	0.126E-03	0.126E-03	0.284E-04	0.284E-04	0.340E-04	0.340E-04
10	0.111E-03	0.121E-03	0.295E-03	0.110E-03	0.335E-04	0.290E-04	0.545E-04	0.340E-04
20	0.141E-03	0.115E-03	0.405E-03	0.148E-03	0.473E-04	0.314E-04	0.211E-03	0.361E-04
30	0.115E-03	0.100E-03	0.322E-03	0.154E-03	0.550E-04	0.272E-04	0.272E-03	0.331E-04
40	0.112E-03	0.967E-04	0.340E-03	0.155E-03	0.943E-04	0.257E-04	0.359E-03	0.374E-04
50	0.112E-03	0.940E-04	0.329E-03	0.164E-03	0.964E-04	0.259E-04	0.325E-03	0.437E-04
70	0.124E-03	0.102E-03	0.578E-03	0.171E-03	0.174E-03	0.258E-04	0.424E-03	0.422E-04
90	0.132E-03	0.105E-03	0.725E-03	0.173E-03	0.254E-03	0.235E-04	0.677E-03	0.422E-04
110	0.154E-03	0.107E-03	0.365E-03	0.164E-03	0.375E-03	0.238E-04	0.753E-03	0.383E-04
130	0.151E-03	0.117E-03	0.289E-03	0.160E-03	0.465E-03	0.226E-04	0.109E-02	0.359E-04
150	0.166E-03	0.127E-03	0.324E-03	0.150E-03	0.600E-03	0.186E-04	0.119E-02	0.463E-04

by the single-precision IUQR algorithm on the problem of order $m = N - k$. It is interesting to note that the downdating procedure produces substantially more accurate answers faster than the IUQR algorithm even for moderately sized values of k.

5 Conclusion

New algorithms are presented for discrete least squares approximation by algebraic polynomials on the unit circle and by trigonometric polynomials. The algorithms downdate the approximant as well as recurrence coefficients for Szegö polynomials when a node-abscissa pair is removed from the inner product. The schemes are based on the QR algorithm for unitary Hessenberg matrices. A particularly attractive application is to combine our algorithm for trigonometric polynomials with the fast Fourier transform algorithm. This yields a fast scheme for computing trigonometric least squares approximants when the nodes of the inner product form a subset of equidistant nodes on $[0, 2\pi)$.

Table 2: 300 nodes with equispaced arguments in $[0, 2\pi)$; random weights in $(0, 10)$

	nodes in order of increasing argument.				nodes in random order.			
	relative error in $\tilde{a}_m$		error in $\tilde{g}_m$		relative error in $\tilde{a}_m$		error in $\tilde{g}_m$	
k	DD	UD	DD	UD	DD	UD	DD	UD
0	0.207E-03	0.207E-03	0.244E-03	0.244E-03	0.511E-04	0.511E-04	0.572E-04	0.572E-04
10	0.206E-03	0.183E-03	0.335E-03	0.181E-03	0.563E-04	0.537E-04	0.844E-04	0.744E-04
20	0.199E-03	0.171E-03	0.417E-03	0.192E-03	0.629E-04	0.512E-04	0.140E-03	0.722E-04
30	0.187E-03	0.175E-03	0.362E-03	0.198E-03	0.785E-04	0.518E-04	0.252E-03	0.750E-04
40	0.188E-03	0.179E-03	0.340E-03	0.203E-03	0.873E-04	0.509E-04	0.303E-03	0.739E-04
50	0.181E-03	0.167E-03	0.334E-03	0.209E-03	0.308E-03	0.487E-04	0.414E-03	0.714E-04
70	0.208E-03	0.169E-03	0.604E-03	0.217E-03	0.406E-03	0.479E-04	0.624E-03	0.638E-04
90	0.192E-03	0.145E-03	0.765E-03	0.222E-03	0.572E-03	0.442E-04	0.178E-02	0.688E-04
110	0.198E-03	0.155E-03	0.389E-03	0.220E-03	0.874E-03	0.333E-04	0.184E-02	0.533E-04
130	0.201E-03	0.155E-03	0.341E-03	0.213E-03	0.894E-03	0.286E-04	0.139E-02	0.401E-04
150	0.201E-03	0.160E-03	0.379E-03	0.195E-03	0.122E-02	0.256E-04	0.153E-02	0.285E-04

Table 3: 300 nodes with equispaced arguments in $[0, \pi)$; uniform weights.

	nodes in order of increasing argument.				nodes in random order.			
	relative error in $\tilde{a}_m$		error in $\tilde{g}_m$		relative error in $\tilde{a}_m$		error in $\tilde{g}_m$	
k	DD	UD	DD	UD	DD	UD	DD	UD
0	0.398E-03	0.398E-03	0.742E-03	0.742E-03	0.125E-03	0.125E-03	0.116E-03	0.116E-03
10	0.455E-03	0.398E-03	0.188E-02	0.743E-03	0.125E-02	0.101E-03	0.106E-02	0.124E-03
20	0.437E-03	0.389E-03	0.187E-02	0.720E-03	0.144E-02	0.112E-03	0.146E-02	0.122E-03
30	0.452E-03	0.390E-03	0.228E-02	0.714E-03	0.170E-02	0.110E-03	0.163E-02	0.131E-03
40	0.479E-03	0.391E-03	0.265E-02	0.694E-03	0.195E-02	0.947E-04	0.167E-02	0.113E-03
50	0.508E-03	0.391E-03	0.271E-02	0.680E-03	0.215E-02	0.886E-04	0.165E-02	0.109E-03
70	0.576E-03	0.431E-03	0.278E-02	0.645E-03	0.251E-02	0.796E-04	0.210E-02	0.109E-03
90	0.533E-03	0.450E-03	0.168E-02	0.605E-03	0.367E-02	0.849E-04	0.302E-02	0.106E-03
110	0.558E-03	0.435E-03	0.193E-02	0.552E-03	0.458E-02	0.795E-04	0.367E-02	0.102E-03
130	0.650E-03	0.423E-03	0.210E-02	0.505E-03	0.528E-02	0.792E-04	0.539E-02	0.840E-04
150	0.110E-02	0.493E-03	0.356E-02	0.466E-03	0.671E-02	0.548E-04	0.528E-02	0.895E-04

Table 4: 300 nodes with random arguments in $[0, 2\pi)$; uniform weights.

	relative error in $\tilde{a}_m$		error in $\tilde{g}_m$	
k	DD	UD	DD	UD
0	0.411E-03	0.111E-03	0.925E-03	0.925E-03
9	0.508E-03	0.109E-03	0.914E-03	0.838E-03
10	0.248E-01	0.160E-03	0.318E-01	0.838E-03
20	0.212E-01	0.129E-03	0.373E-01	0.789E-03
30	0.220E-01	0.164E-03	0.328E-01	0.735E-03
40	0.114E-01	0.117E-03	0.131E-01	0.826E-03
50	0.186E-01	0.124E-03	0.240E-01	0.654E-03
70	0.196E-01	0.137E-03	0.264E-01	0.658E-03
90	0.210E-01	0.122E-03	0.275E-01	0.483E-03
110	0.347E-01	0.137E-03	0.820E-01	0.100E-02
130	0.502E-01	0.120E-03	0.852E-01	0.508E-03
150	0.513E-01	0.788E-04	0.865E-01	0.360E-03

Table 5: Downdating the FFT

$N = 1024$		relative error in $\tilde{a}_m$		error in $\tilde{g}_m$		CPU seconds	
m	k	DD	UD	DD	UD	DD	UD
1013	11	0.461E-05	0.213E-01	0.218E-05	0.204E-01	0.303E+01	0.148E+03
993	31	0.502E-05	0.215E-01	0.623E-05	0.195E-01	0.814E+01	0.142E+03
973	51	0.603E-05	0.217E-01	0.142E-04	0.193E-01	0.132E+02	0.136E+03
923	101	0.900E-05	0.193E-01	0.257E-04	0.244E-01	0.253E+02	0.122E+03
913	111	0.944E-05	0.191E-01	0.265E-04	0.242E-01	0.276E+02	0.119E+03
893	131	0.111E-04	0.178E-01	0.312E-04	0.219E-01	0.323E+02	0.114E+03
873	151	0.129E-04	0.174E-01	0.373E-04	0.220E-01	0.368E+02	0.109E+03
853	171	0.143E-04	0.172E-01	0.462E-04	0.221E-01	0.412E+02	0.104E+03
833	191	0.172E-04	0.174E-01	0.566E-04	0.220E-01	0.456E+02	0.990E+02
813	211	0.223E-04	0.171E-01	0.721E-04	0.241E-01	0.497E+02	0.941E+02
713	311	0.477E-04	0.133E-01	0.131E-03	0.169E-01	0.692E+02	0.719E+02
613	411	0.167E-03	0.104E-01	0.158E-03	0.275E-01	0.862E+02	0.527E+02
513	511	0.267E-03	0.769E-02	0.527E-03	0.174E-01	0.101E+03	0.365E+02
413	611	0.286E-03	0.417E-02	0.487E-03	0.852E-02	0.112E+03	0.233E+02
313	711	0.337E-03	0.322E-02	0.554E-03	0.781E-02	0.122E+03	0.132E+02

$N = 2048$		relative error in $\tilde{a}_m$		error in $\tilde{g}_m$		CPU seconds	
m	k	DD	UD	DD	UD	DD	UD
2037	11	0.598E-05	0.105E+00	0.259E-05	0.184E+00	0.391E+01	0.390E+03
1997	51	0.740E-05	0.110E+00	0.151E-04	0.235E+00	0.172E+02	0.374E+03
1947	101	0.100E-04	0.102E+00	0.293E-04	0.164E+00	0.336E+02	0.355E+03
1897	151	0.125E-04	0.980E-01	0.426E-04	0.126E+00	0.495E+02	0.337E+03
1847	201	0.159E-04	0.918E-01	0.642E-04	0.131E+00	0.649E+02	0.319E+03
1797	251	0.237E-04	0.881E-01	0.812E-04	0.136E+00	0.799E+02	0.301E+03
1747	301	0.289E-04	0.865E-01	0.116E-03	0.157E+00	0.946E+02	0.285E+03
1697	351	0.358E-04	0.834E-01	0.115E-03	0.137E+00	0.109E+03	0.268E+03
1647	401	0.514E-04	0.809E-01	0.170E-03	0.125E+00	0.123E+03	0.252E+03
1597	451	0.749E-04	0.748E-01	0.212E-03	0.133E+00	0.136E+03	0.237E+03
1547	501	0.779E-04	0.729E-01	0.243E-03	0.132E+00	0.149E+03	0.222E+03
1447	601	0.953E-04	0.695E-01	0.313E-03	0.149E+00	0.174E+03	0.194E+03
1347	701	0.242E-03	0.628E-01	0.396E-03	0.114E+00	0.197E+03	0.167E+03
1147	901	0.517E-03	0.443E-01	0.109E-02	0.116E+00	0.238E+03	0.121E+03
1047	1001	0.731E-03	0.381E-01	0.113E-02	0.640E-01	0.256E+03	0.100E+03
947	1101	0.114E-02	0.285E-01	0.183E-02	0.523E-01	0.273E+03	0.815E+02

References

[1] G.S. Ammar, W.B. Gragg and L. Reichel, Constructing a unitary Hessenberg matrix from spectral data, in *Numerical Linear Algebra, Digital Signal Processing and Parallel Algorithms*, eds. G.H. Golub and P. Van Dooren, Springer, New York, 1991, pp. 385–396.

[2] A. Björck and V. Pereyra, Solution of Vandermonde systems of equations, *Math. Comp.* 24:893–903 (1970).

[3] S. Elhay, G.H. Golub and J. Kautsky, Updating and downdating of orthogonal polynomials with data fitting applications, *SIAM J. Matrix Anal. Appl.* 12:327–353 (1991).

[4] G.H. Golub and J.H. Welsch, Calculation of Gauss quadrature rules, *Math. Comp.* 23:221–230 (1969).

[5] W.B. Gragg, The QR algorithm for unitary Hessenberg matrices, *J. Comput. Appl. Math.* 16:1–8 (1986).

[6] W.B. Gragg and W.J. Harrod, The numerically stable reconstruction of Jacobi matrices from spectral data, *Numer. Math.* 44:317–335 (1984).

[7] L. Reichel, Fast QR decomposition of Vandermonde-like matrices and polynomial least squares approximation, *SIAM J. Matrix Anal. Appl.* 12:552–564 (1991).

[8] L. Reichel, G.S. Ammar and W.B. Gragg, Discrete least squares approximation by trigonometric polynomials, *Math. Comp.* 57:273-289 (1991).

[9] L.B. Scott and L.R. Scott, Efficient methods for data smoothing, *SIAM J. Numer. Anal.* 26:681–692 (1989).

DISTRIBUTION LIST

DIRECTOR (2)
DEFENSE TECH. INFORMATION
CENTER, CAMERON STATION
ALEXANDRIA, VA 22314

DIRECTOR OF RESEARCH ADMIN.
CODE 81
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CA 93943

LIBRARY (2)
CODE 52
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CA 93943

NATIONAL SCIENCE FOUNDATION
WASHINGTON, D.C. 20550

DEPARTMENT OF MATHEMATICS
CODE MA
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CA 93943

PROFESSOR WILLIAM GRAGG (20)
CODE MA/GR
DEPARTMENT OF MATHEMATICS
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CA 93943